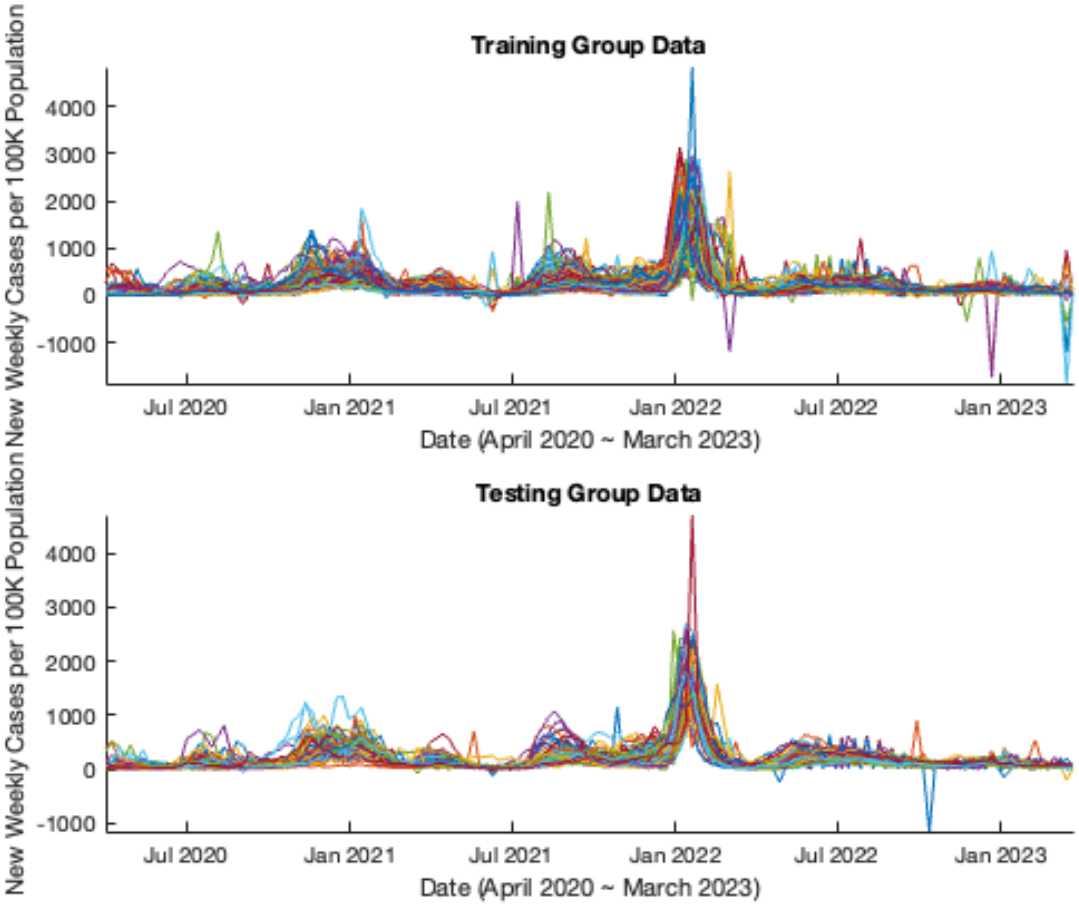
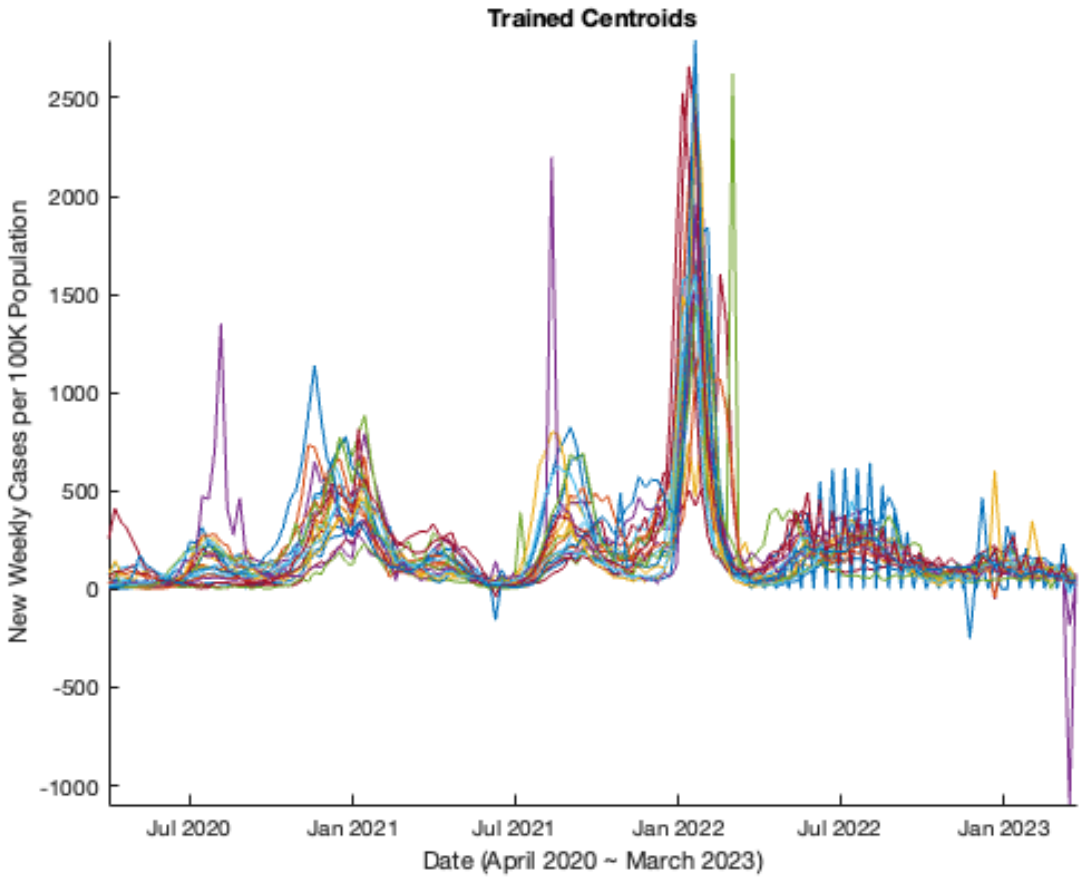
```
load('COVIDbyCounty.mat');

training = []; % table of all locations for training stage
testing = []; % table of all locations for testing stage
trainingCases = []; % 156-week COVID data for training locations
testingCases = []; % 156-week COVID data for testing locations

% 3-1 ratio of training group and testing group; in other words, 75% of all
% locations will be implemented for training & the remaining 25% for
% testing
for i = 1:(length(CNTY_CENSUS.fips))
    if mod(i, 4) == 0 % assign to testing for every 4 locations
        testing = [testing; CNTY_CENSUS(i, :)];
    else % assign the remaining three quarters to training
        training = [training; CNTY_CENSUS(i, :)];
    end
end

for i = 1:height(training) % index from CNTY_COVID (original raw data)
    trainingCases = [trainingCases; CNTY_COVID(CNTY_CENSUS.fips == ...
        training(i, :).fips, :)];
end

for i = 1:height(testing)
    testingCases = [testingCases; CNTY_COVID(CNTY_CENSUS.fips == ...
        testing(i, :).fips, :)];
end

split = tiledlayout(2, 1);
nexttile;
hold on;
for i = 1:height(trainingCases)
    plot(dates, trainingCases(i, :));
end
hold off;
axis tight;
title('Training Group Data');
xlabel('Date (April 2020 ~ March 2023)');
ylabel('New Weekly Cases per 100K Population');
nexttile;
hold on;
for i = 1:height(testingCases)
    plot(dates, testingCases(i, :));
end
hold off;
axis tight;
title('Testing Group Data');
xlabel('Date (April 2020 ~ March 2023)');
ylabel('New Weekly Cases per 100K Population');
exportgraphics(split, 'training_testing_group_data.png');

k = randi([9, 25], 1, 1); % k clusters
```

```
% apply k-means algorithm on training group
[indices, centroids] = kmeans(trainingCases, k, 'Replicates', 10);

figure;
hold on;
for i = 1:height(centroids)
    plot(dates, centroids(i, :));
end
hold off;
axis tight;
title('Trained Centroids');
xlabel('Date (April 2020 ~ March 2023)');
ylabel('New Weekly Cases per 100K Population');
exportgraphics(gca, 'trained_centroids.png');

centroidDivision = [];
centroid_labels = [];

for i = 1:k % loop through the k centroids
    for j = 1:height(indices)
        if indices(j) == i
            centroidDivision = [centroidDivision, training.DIVISION(j)];
        end
    end
    % determine the most frequent division in each centroid in order to
    % determine which division said centroid belongs to
    centroid_labels = [centroid_labels; mode(centroidDivision)];
    centroidDivision = []; % clear current row for the next row of entries
end
```


Training Group Data
New Weekly Cases per 100K Population
4000
3000
2000
1000
0
-1000
Jul 2020
Jan 2021
Jul 2021
Jan 2022
Jul 2022
Jan 2023
Date (April 2020 ~ March 2023)
Testing Group Data
New Weekly Cases per 100K Population
4000
3000
2000
1000
0
-1000
Jul 2020
Jan 2021
Jul 2021
Jan 2022
Jul 2022
Jan 2023
Date (April 2020 ~ March 2023)



Published with MATLAB® R2023a